(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平11－7451

(43)公開日　平成11年(1999)１月12日

(51)Int.Cl.[6]	識別記号	ＦＩ		
Ｇ０６Ｆ　17/30		Ｇ０６Ｆ	15/419	３１０
			15/40	３７０Ｊ

審査請求　未請求　請求項の数９　ＯＬ　（全 22 頁）

(21)出願番号　特願平9－161458

(22)出願日　平成９年(1997)６月18日

(71)出願人　000005496
富士ゼロックス株式会社
東京都港区赤坂二丁目17番22号

(72)発明者　梅基　宏
神奈川県足柄上郡中井町境430　グリーンテクなかい　富士ゼロックス株式会社内

(72)発明者　舘野　昌一
神奈川県足柄上郡中井町境430　グリーンテクなかい　富士ゼロックス株式会社内

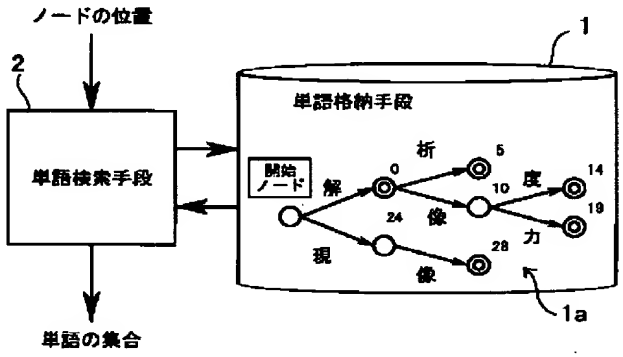
(74)代理人　弁理士　服部　毅巖

(54)【発明の名称】　単語検索装置及び単語検索プログラムを記録した媒体

(57)【要約】

【課題】　少ない記憶容量で高速に単語の検索が行えるようにする。

【解決手段】　単語格納手段１には、単語の集合１ａが、深さ優先順にノードが記録されるトライ形式（トライ・インデックス）で格納されている。単語検索手段２は、単語格納手段１における格納先の位置情報が入力されると、単語格納手段１のトライを根から順にたどっていき、入力された位置のノードをまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語集合を出力する。これにより、ノード位置から、そのノードを含む単語の集合を得ることができる。しかも、トライ中のノードを指定されたときに、親のノードへのリンク情報を用いることなく、そのノードを含む経路を特定することができる。



1

【特許請求の範囲】

【請求項１】　単語集合から単語を検索する単語検索装置において、

深さ優先順にノードが記録されるトライ形式にしたがって、ノードに対応付けられた単語の集合が格納された単語格納手段と、

前記単語格納手段におけるノードの位置が入力されると、前記単語格納手段のトライを根から順にたどっていき、入力された位置のノードまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語の集合を出力する単語検索手段と、

を有することを特徴とする単語検索装置。

【請求項２】　前記単語格納手段に含まれる単語に対応するキーと、各単語を構成しているノードの位置とを対応付けて格納するキーインデックス格納手段と、

前記キーインデックス格納手段中の任意のキーが入力されると、前記キーインデックス格納手段から、入力されたキーに対応するノードの位置の集合を取得し、取得したノードの位置の集合を前記単語検索手段に対して出力するノード位置検索手段と、

をさらに有することを特徴とする請求項１記載の単語検索装置。

【請求項３】　前記キーインデックス格納手段は、前記単語格納手段に含まれる単語を構成する全ての文字と、各文字を表しているノードの位置とを対応付けていることを特徴とする請求項２記載の単語検索装置。

【請求項４】　前記キーインデックス格納手段は、前記単語格納手段に含まれる単語を構成する文字のうち、単語の先頭文字および末尾文字を除いたすべての文字と、各文字を表しているノードの位置とを対応付けていることを特徴とする請求項３記載の単語検索装置。

【請求項５】　正規表現が入力されると、入力された正規表現を解析し、前記ノード位置検索手段に対し、正規表現中の文字を渡してノードの位置の集合を受け取ると共に、前記単語検索手段に対し、ノードの位置の集合を渡して単語の集合を受け取り、入力された正規表現に合致する単語の集合を出力する正規表現解析手段を更に有することを特徴とする請求項３記載の単語検索装置。

【請求項６】　前記単語格納手段は、単語と各単語に対応する情報へのポインタとを組にして格納しており、前記単語検索手段は、前記単語格納手段から各々のノードが表す単語および単語に対応する情報へのポインタの集合を出力することを特徴とする請求項１記載の単語検索装置。

【請求項７】　前記単語格納手段は、深さ優先順にノードが記録されるトライ形式にしたがって、ノードに対応付けられた単語の集合を格納していると共に、単語と各単語に関連する関連単語のノードの位置の集合とを対応付けて格納しており、

2

単語が入力されると、前記単語格納手段から、入力された単語に関連する単語のノードの位置の集合を取得し、取得したノードの位置の集合を前記単語検索手段に対して出力するノード位置検索手段を更に有することを特徴とする請求項１記載の単語検索装置。

【請求項８】　前記単語格納手段は、表記によって表された表記単語の集合を、深さ優先順にノードが記録されるトライ形式で格納していると共に、表記単語と各表記単語に対応するの文字列を構成しているよみ単語のノードの位置の集合とを対応付けて格納するよみインデックス格納手段と、よみによって表された単語の集合を、深さ優先順にノードが記録されるトライ形式で格納していると共に、よみ単語と各よみ単語に対応する表記単語の文字列を構成しているノードの位置の集合とを対応付けて格納している表記インデックス格納手段と、から構成されていることを特徴とする請求項７記載の単語検索装置。

【請求項９】　コンピュータに単語集合から単語を検索させるための単語検索プログラムを記録した媒体において、

深さ優先順にノードが記録されるトライ形式にしたがって、ノードに対応付けられた単語の集合が格納された単語格納手段、

前記単語格納手段におけるノードの位置が入力されると、前記単語格納手段のトライを根から順にたどっていき、入力された位置のノードをまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語の集合を出力する単語検索手段、

としてコンピュータを機能させるための単語検索プログラムを記録した媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は単語の集合の中から単語につけられたキーをもとに効率的に単語を検索する単語検索装置及び単語検索プログラムを記録した媒体に関し、特に任意の位置にある文字を指定して、関連する単語を高速に検索する単語検索装置及びそのような検索をコンピュータに行わせるための単語検索プログラムを記録した媒体に関する。

【０００２】

【従来の技術】何らかのキーから単語を検索するという処理は、辞書の検索やかな漢字変換といったテキスト情報処理システムにおける基本的な過程である。それだけに、キーから単語を検索するために必要な処理速度や記憶容量は、そのような処理システム全体の性能を大きく左右する。したがって、このような処理をより高速に、かつより少ない記憶容量から実現することによって、非常に大きな実用的効果を得ることができる。

【０００３】さて、単語をキーとして文書を検索するシ

ステムはすでに多く存在しているが、このような文書検索システムにおいても、検索の結果として単語を出力する機能は、必須ではないが非常に有効な場合がある。

【０００４】文書検索システムにおいて多くの場合、キーである単語は、文書中に含まれているか、もしくはシステムの管理者や文書の作成者によってあらかじめ選定される。この場合、検索システムを利用する立場のユーザからは、登録してあるキーの単語が何であるか分からないことがしばしばある。そのため、ユーザを支援するために、文書ではなくキーワードを何らかの方法で検索できることが必要となる。

【０００５】キーワードを検索する具体的な方法として、たとえば、よみから表記の単語を検索する方法、または、ある文字もしくは文字列を含む単語を検索する方法、さらには、任意の正規表現を満たす単語を検索する方法などが考えられる。

【０００６】また、キーワードによる文書検索システムにおいては、単語から所望の文書もしくは文書へのポインタを高速に得るために、インデックスとしてトライと呼ばれる木構造（トライ・インデックス）を用いることが多い。このトライを用いれば、高速に単語の検索を行うことができる。トライから単語を検索するときには、ほぼ入力文字列の長さに比例する程度の処理ステップ数しか必要としない。またデータ圧縮率も比較的良いため、トライは大量の索引単語を格納するという用途に向いている。加えて、トライを用いる場合、単語の先頭部分の文字列を指定すると、その文字列から始まるすべての単語を、簡単な処理によって求めることができるという利点もある。

【０００７】ところが、トライから、単語の先頭以外の任意の位置の部分文字を含む単語を、高速に探し出すことはできない。ここで、これらの場合には、単語を表すトライとは別個にインデックスを設けることによって、任意の位置の部分文字から、その文字を含む単語を検索することが行われている。

【０００８】いま、文字を検索キーとして、その文字から綴りの中の任意の位置に含む単語を検索する場合を考えることにしよう。単語は複数の文字によって構成されているので、ある１つの単語は、複数の検索キー（文字）に対応付けられている。この場合のように、検索対象が複数のキーにリンクされているときは、検索対象の集合を１つのデータ構造の中に格納し、検索キーにはそのデータ構造中のポインタの値を対応させると、必要な記憶容量は少なくて済む。

【０００９】そこで、単語集合のデータを少ない記憶容量で表すことができ、かつ、ポインタによって単語を特定できるデータ構造について以下に考察する。単語の集合を格納するデータ構造として一般的なのは、固定長または可変長の文字列として格納するレコード構造である。このレコード構造を用いた単語検索方式を「第１の

従来例」と呼ぶことにする。このレコード構造であれば、任意の単語に対して、単語の総数によらずほぼ一定の時間で高速にアクセスすることができる。したがって、単語集合をこのようなレコード構造として格納することによって、単語検索システムにおける高速なキーワード検索を実現することができる。

【００１０】また、第１の従来例とは別に、トライ・インデックスを単語集合のデータとみなして、単語の末尾に対応するトライ中のノードの識別番号を単語へのポインタとすることが考えられる。これを「第２の従来例」と呼ぶことにする。トライのような木構造においては、根を除くすべてのノードの親ノードは一意に存在する。したがって、親ノードへのリンク情報をすべてのノードに持たせることによって、１つのノードを指定したときに、そのノードから根に至るまでの経路は一意に決定することができる。

【００１１】

【発明が解決しようとする課題】しかし、単語をキーとするトライ・インデックスが存在し、なおかつそのインデックス中の単語を何らかのキーから検索する場合において、上記の従来の方式には以下のような問題点があった。

【００１２】上記の第１の従来例、すなわち、単語の集合を固定長または可変長の文字列として格納するレコード構造のデータを、インデックスとは別個に用意する方法では、キーから単語への高速な検索は実現できるが、すでにトライ形式で格納されている単語の集合のデータとは別個にあらたなデータが必要となる。そのために必要な記憶容量は無視し得ないほど大きくなってしまうという問題点がある。

【００１３】上記の第２の従来例、すなわち、トライ・インデックスを単語集合のデータとみなして、単語の末尾に対応するトライ中のノードの識別番号を単語へのポインタとする方法では、第１の従来例と比べて単語集合を表すためのデータが不要な分だけ、キーから単語への検索自体に必要な記憶容量は第１の従来例に比べて少なくて済むが、トライ・インデックスに対して本来ならば不要な、親ノードへのリンク情報を追加することになってしまう。したがって、検索システム全体としてみた場合、第２の従来例は第１の従来例に比べて、記憶容量の面で著しく改善されているとは言えない。

【００１４】本発明はこのような点に鑑みてなされたものであり、少ない記憶容量で高速に単語を検索できる単語検索装置を提供することを目的とする。また、本発明の他の目的は、コンピュータに対して、少ない記憶容量で高速に単語を検索させるための単語検索プログラムを記録した媒体を提供することである。

【００１５】

【課題を解決するための手段】本発明では上記課題を解決するために、単語集合から単語を検索する単語検索装

置において、深さ優先順にノードが記録されるトライ形式にしたがって、ノードに対応付けられた単語の集合が格納された単語格納手段と、前記単語格納手段におけるノードの位置が入力されると、前記単語格納手段のトライを根から順にたどっていき、入力された位置のノードをまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語の集合を出力する単語検索手段と、を有することを特徴とする単語検索装置が提供される。

【００１６】この単語検索装置によれば、単語格納手段における格納先の位置情報が入力されると、単語検索手段が、単語格納手段のトライを根から順にたどっていき、入力された位置のノードまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語の集合が出力される。その結果、入力された位置に対応する単語を含む複数の単語を、高速に検索することができるとともに、単語格納手段に必要な記憶容量は少なくてすむ。

【００１７】また、コンピュータに単語集合から単語を検索させるための単語検索プログラムを記録した媒体において、深さ優先順にノードが記録されるトライ形式にしたがって、ノードに対応付けられた単語の集合が格納された単語格納手段、前記単語格納手段における格納先の位置情報が入力されると、前記単語格納手段のトライを根から順にたどっていき、入力された位置のノードをまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語の集合を出力する単語検索手段、としてコンピュータを機能させるための単語検索プログラムを記録した媒体が提供される。

【００１８】この媒体に記録された単語検索プログラムをコンピュータに実行させることにより、深さ優先順にノードが記録されるトライ形式にしたがって、ノードに対応付けられた単語の集合が格納された単語格納手段としての機能と、単語格納手段における格納先の位置情報が入力されると、単語格納手段のトライを根から順にたどっていき、入力された位置のノードをまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語の集合を出力する単語検索手段としての機能とが、コンピュータによって実現される。

【００１９】

【発明の実施の形態】以下、本発明の実施の形態を図面を参照して説明する。図１は、本発明の原理構成図である。本発明に係る単語検索装置は、単語格納手段１と単語検索手段２とを有している。

【００２０】単語格納手段１には、単語の集合１ａが、深さ優先順にノードが記録されるトライ形式（トライ・インデックス）で格納されている。ここで、ノードの深さとは、トライの根からそのノードまでの経路の長さである。そして、ノードの深さ優先順とは、できる限り探索を深さ方向に進めたときにたどるノードの順番である。あるノードの子孫のノードと、弟のノードとを比較した場合、常に、子孫のノードの方が優先順が高い（アドレスの値が小さい）。また、兄弟同士で比べると、兄となるノードの方が優先順が高い。

【００２１】単語検索手段２は、単語格納手段１における格納先の位置情報が入力されると、単語格納手段１のトライを根から順にたどっていき、入力された位置のノードまでの経路を求め、求められた経路以降の全ての経路をたどって到達するノードに対応する全ての単語を取得し、取得した単語集合を出力する。ノードの位置からそのノードを含む単語または単語の集合を求めるアルゴリズムを以下に記述する。

【００２２】図２は、ノードの位置からそのノードを含む単語または単語の集合を求めるアルゴリズムを示すフローチャートである。これは、ノードの位置情報を受け取った単語検索手段２が行う処理である。

［Ｓ１］単語格納手段１のトライにおいて、開始ノードから遷移することを考える。まず、トライにおけるラベルの列を格納するために空のラベルスタックを用意し、開始ノードを「ノードＡ」とおき、与えられたノード位置に相当するノードを「ノードＸ」とおく。

［Ｓ２］ノードＡの長子のノードを「Ｂ１」とする。ノードＡの次の位置に存在するため、長子のノードであるノードＢ１は直ちに求まる。

［Ｓ３］ノードＢ１の隣接する弟ノードをノードＢ２とする。トライを根からたどることによって、任意のノードの隣接する弟のノードＢ２も直ちに求まる。

［Ｓ４］ノードＸがノードＢ１と等しいか否かを判断する。等しければステップＳ８に進み、等しくなければステップＳ５に進む。

［Ｓ５］ノードＸがノードＢ２よりも前か否かを判断する。前であるならステップＳ６に進み、前でないならステップＳ６に進む。

［Ｓ６］ノードＸがノードＢ２よりも前にある場合、ノードＸは、ノードＢ１までの経路をたどることが分かる。そこで、ノードＡからノードＢ１に至るまでのアークに付加されたラベルをラベルスタックに格納し、ノードＡにノードＢ１を代入して、ステップＳ２に戻る。

［Ｓ７］ノードＸがノードＢ２以降にある場合、ノードＸが、ノードＢ１の経路をたどらないことが分かる。そこで、ノードＢ１にノードＢ２を代入して、ステップＳ３に戻る。

［Ｓ８］ノードＸがノードＢ１と等しい場合、ノードＢ１以降の全ての経路をたどり、それらの経路となるアークに付加されたラベル列を求める。

［Ｓ９］ラベルスタックに格納されているラベル列とノードＢ１以降の経路から得られたラベル列の連結し、処理を終了する。これにより、求めるべき単語が表され

る。

【００２３】このような処理を単語検索手段２が行うことにより、ノード位置から、そのノードを含む単語の集合を得ることができる。しかも、トライ中のノードが深さ優先順に記録され、かつトライを根からたどることによって任意のアークに関してその隣接する弟アークが特定できるため、トライ中のノードを指定されたときに、親のノードへのリンク情報を用いることなく、そのノードを含む経路を特定することができる。その理由を以下に説明する。

【００２４】まず、前提として、トライ中のあるノードをノードＡとし、ノードＡの長子のノードをＢ１とし、ノードＢ１の隣接する弟ノードをＢ２とし、ノードＢ１の長子のノードをＣとする。いま、ノードＸの位置が指定され、ノードＡからノードＸに至るまで経路を求めたいとする。このとき、ノードＸはノードＡからたどれることは分かっているものとする。

【００２５】本発明に係るトライでは、ノードは深さ優先順に記録されているので、ノードＣは、ノードＢ１とノードＢ２との間にある。仮に、ノードＸがノードＣと等しいとき、ノードＸは、ノードＢ１よりも後にあり、かつ、ノードＢ２よりも前にあることになる。したがって、ノードＸはノードＢ１からたどれることになる。以上のことから、ノードＸがノードＢ１とノードＢ２との間にある場合、ノードＸにはノードＡからノードＢ１に遷移する経路をたどって到達できることがわかる。

【００２６】また、ノードＸがノードＢ２よりも後にある場合、ノードＸは、ノードＡからノードＢ１に遷移する経路をたどらないことがわかる。この場合、ノードＸにはノードＡから、ノードＢ１の弟のノードのいずれかを遷移する経路をたどって到達できることが分かる。以上の考察を繰り返すことによって、ノードＡからノードＸまで到達できる経路が分かる。

【００２７】以上の結果、ノードの位置を指定されたときに、トライの根から遷移することによって、そのノードを含む単語を求めることができる。しかも、トライにおける単語を、トライにおけるノードの位置をポインタとして参照することによって、トライ・インデックス以外に単語集合を表すデータを設けておく必要がなく、情報の記憶容量が少なくてすむ。

【００２８】また、単語に対する検索キー（文字）の入力に応じて、単語の集合が出力されるようにすることもできる。そのような単語検索装置について、以下に説明する。

【００２９】図３は、単語へのキーを入力とする単語検索装置の原理構成図である。この単語検索装置は、単語格納手段１１、キーインデックス格納手段１２、ノード位置検索手段１３及び単語検索手段１４で構成されている。なお、単語格納手段１１と単語検索手段１４とは、図１中の単語格納手段１と単語検索手段２と同じ機能有

しているため、ここでは説明を省略する。

【００３０】キーインデックス格納手段１２は、単語へのキー（文字）と、そのキーと同じラベルが付加されているノードの位置情報とを対応付けて格納している。このように、キーとノード位置との対応関係のすべてをもとにキーインデックス格納手段１２が構成されている。なお、１つのキーに対して複数の単語が対応している場合、キーインデックス格納手段１２において、そのキーにはノードの位置の集合が対応している。

【００３１】ノード位置検索手段１３は、単語へのキーが入力されると、キーインデックス格納手段１２の中から、対応するノードの位置の集合を検索する。そして、得られたノードの位置の集合を単語検索手段１４へ入力する。

【００３２】以後、単語検索手段１４が、単語格納手段１１から各ノード位置に対応する単語集号を検索し、単語集合を出力する。これにより、単語を構成している文字を検索キーとして入力し、その文字を含む単語の集合を得ることができる。すなわち、単語の先頭以外の任意の位置の部分文字を含む単語の集合を探し出すことができる。

【００３３】なお、キーインデックス格納手段１２は、単語格納手段１１に含まれる単語を構成する文字のうち、単語の先頭文字および末尾文字を除いたすべての文字と、単語格納部中でのその文字を表しているノードの位置とを対応付けているものであってもよい。それは、単語の先頭文字あるいは末尾文字から始まる単語は、従来の技術を用いて検索できるため、特にキーインデックス格納手段１２において管理する必要がないからである。このように、単語の先頭文字および末尾文字に対応するノードの位置の情報を省略すれば、キーインデックス格納手段１２に必要な記憶容量をさらに減らすことができる。

【００３４】また、単語格納手段１１には、単語に対応する情報（例えば、その単語を含む文書）へのポインタとそれに対応する単語とを組にして格納しておき、単語検索手段１４は、単語格納手段１１から各々のノードが表す単語および単語に対応する情報へのポインタの集合を出力するようにしてよい。これにより、任意のキー（文字）を入力することにより、そのキーを含む単語を取得し、さらに、取得した単語に対応する情報（文書など）を得ることができる。

【００３５】以上が本発明の基本となる原理構成である。以下に、上記の構成をより具体化した単語検索装置の実施の形態を説明する。図４は、本発明の第１の実施の形態を示すブロック図である。これは、単語を入力として、その単語に関連する単語の集合を検索する単語検索装置である。

【００３６】この実施の形態に係る単語検索装置は、関連単語インデックス部２１、ノード位置検索部２２及び

単語検索部２３で構成されている。関連単語インデックス部２１には、図３の原理構成で示した単語格納手段１１とキーインデックス格納手段１２との情報を保持しているとともに、ある単語とその単語に対する関連語集合との対応関係をも保持している。

【００３７】ノード位置検索部２２は、単語の入力を受けると、その単語に関連する単語（その単語自身も含む）のノードの位置の集合を受け取り、単語検索部２３に入力する。単語検索部２３は、入力されたノードの位置の集合に基づいて、関連単語インデックス部２１を検索し、関連単語の集合を取得し、出力する。

【００３８】ここで、本発明の実施の形態の詳細を説明する前に、木に関する用語をあらためて定義する。木は、ノードとよばれる要素の集合に対して階層関係を与えたものである。以下、木において与えられる階層関係は、親子親戚関係を表すことばで表現することにする。木においては、自らを含むすべてのノードを子孫とするノードが１つ存在する。これを開始ノードとよぶことにする。開始ノード以外のすべてのノードに対して、その親であるノードが必ず１つ存在する。ノードとノードとの間の親子関係のつながりを示すものを、アークとよぶことにする。そして、ノードは、特別な状態として、終了状態を持つことができることにし、終了状態ではない他のノードと区別することにする。また、自分自身以外の子孫を持たないノードは、終了状態であるとする。

【００３９】トライは木構造の一種であり、開始ノードから終了状態のノードまでを、任意個のノードを経由してアークによって結ばれている経路の１本１本に、集合の中の１つの単語が対応している。本説明文中では、トライにおいて、単語を構成する要素である文字を、アークに対して割りつけることにする。また、１つのノードから派生している各々のアークに対応する文字は、すべて異なるようにノードを構成する。トライにおいては、開始ノードを除いて１つのノードに遷移するアークは必ず１つ存在するので、このようなノードとアークの組を１つの辺節という単位と見なすことにする。以下にある単語の集合を表したトライの例を示す。

【００４０】図５は、単語の集合の例を示す図である。この単語集合には、６個の単語「解」「解析」「解像度」「解像力」「現像」「像」がある。それぞれの単語には、ポインタが対応付けられている。ポインタは、その単語を含む文書の識別子集合の位置を指し示すものであるが、この実施の形態では、関連語のノード位置の集合を指し示すのにも用いられる。「解」のポインタは「Ｔ１」であり、「解析」のポインタは「Ｔ２」であり、「解像度」のポインタは「Ｔ３」であり、「解像力」のポインタは「Ｔ４」であり、「現像」のポインタは「Ｔ５」であり、「像」のポインタは「Ｔ６」である。この単語の集合を基に、深さ優先順のノードが記録されたトライを生成する。

【００４１】図６は、深さ優先順にノードが記録されたトライの例を示す図である。これは、図５に示した６個の単語「解」「解析」「解像度」「解像力」「現像」「像」をトライによって表したものである。図中、丸印若しくは２重丸で表しているのがノード３０～３８である。２重丸は終了状態のノード（対応する単語が存在するノード）を表している。根であるノード３０が単語検索時の「開始ノード」となる。終了状態のノードの右下にある記号はその終了状態のノードによって示されることばに関連する単語の集合を意味する識別子をそれぞれ表している。なお、図中の開始ノード以外の各ノード３１～３８の近傍に表示しているのが、それぞれのノードのアドレス（位置）である。また、各ノード３１～３８を接続している矢印がアーク４１～４８である。各アーク４１～４８の上にある文字（ここでは漢字１文字）はラベルである。

【００４２】図６におけるトライのノードを深さ優先順に並べると、根のノード３０から派生する「解」のラベルをもつアーク４１を遷移してきたノード３１、ノード３１から派生する「析」のラベルをもつアーク４２を遷移してきたノード３２、ノード３１から派生する「像」のラベルをもつアーク４３を遷移してきたノード３３、…の順となる。

【００４３】このように、辺節を深さ優先順に記録し、かつ、ある辺節からそのすぐとなりの弟の位置が特定できるようにする。図７は、トライ・インデックスの例を示す図である。この図には、辺節の情報が格納されたアドレス、その辺節の直下の弟ノードの位置（アドレス）、その辺節に対応するラベル、ノードの状態及び対応する文書集合へのリンク情報を示している。この例では、ラベルは漢字１文字である。ノードの状態は、「終了」「継続」のいずれか一方、若しくは双方が設定されている。「終了」は、そのノードに対応する単語が存在することを示し、「継続」は、そのノードが子供を有していることを示す。

【００４４】このようにトライを構成することによって、ある「辺節Ａ」が指定されたときに、「辺節Ａ」の長男および「辺節Ａ」のすぐとなりの弟の辺節を直ちに特定できる。

【００４５】具体的な事例データを想定して、本実施例の動作を説明しよう。簡単のために、図５に示す単語の集合が与えられ、そのうち「解像度」と「解像力」が互いに関連しているとする。

【００４６】まず、すべての単語を表すトライを作成し、単語の末尾に相当する最終ノードには、その単語に関連する単語の集合へのリンクを張っておく。これをトライ１とよぶ。図５の単語集合が与えられた場合であれば、図６のトライを構成する。

【００４７】次に、単語の集合へのリンク情報から単語を対応付けるテーブルを用意する。これを関連語対応テ

ーブルとよぶことにする。図８は、関連語対応テーブルを示す図である。このテーブルは、単語から関連語集合へのリンク情報とノード位置の集合とが対応付けられている。この例では、「解像度」と「解像力」が互いに関連していることから、リンク情報「Ｔ３」「Ｔ４」にはノードの位置「１４，１９」が対応している。

【００４８】次に、本実施の形態の検索を実行する動作を、例を交えて説明する。検索キーとして単語が入力され、入力単語に関連する単語集合を求める場合を考えることにする。いま想定している事例データのもとで、「解像度」が検索キーとして入力された場合を考える。

【００４９】まず、ノード位置検索部２２によって、検索キーの単語「解像度」に対応するノード位置の集合、すなわち、関連単語集合へのリンク情報を求められる。具体的には次のように行う。トライから単語を検索する通常の方法によって、トライ・インデックスから「Ｔ３」というリンク情報を求める。そして、関連語対応テーブルから、「Ｔ３」に対応するノード位置集合として、「１４，１９」を得る。

【００５０】次に、単語検索部２３が、ノード位置検索部２２によって得られたノード位置集合から、対応する単語集合を求める。以下に、図６のトライに基づいて、１つのノード位置をキーとして対応する単語を検索する手順を説明する。

【００５１】図９は、第１の実施の形態における単語検索手順を示すフローチャートである。

［Ｓ１１］検索キーのノードアドレスは「Ａ」であたえられたとする。まず、単語の文字列を記録するためのラベル記録部を用意しておく。はじめはトライの先頭にある辺節に注目する。注目している辺節を「現在の辺節」と、その辺節に含まれるアークとノードとを、それぞれ「現在のアーク」、「現在のノード」とよぶことにする。

［Ｓ１２］現在の辺節が「Ａ」と等しいか否かを判断する。等しければステップＳ１３に進み、等しくなければステップＳ１４に進む。

［Ｓ１３］現在の辺節のアドレスが、「Ａ」と等しいとき、現在のアークのラベルをラベル記録部にプッシュ（格納）し、ラベル記憶部の内容を出力し、正常に処理を終了する。

［Ｓ１４］現在の辺節が弟を有しているか否かを判断する。弟を有していればステップＳ１５に進み、有していなければステップＳ１７に進む。

［Ｓ１５］現在の辺節が弟を有している場合には、隣接する弟の辺節のアドレスが「Ａ」以下か否かを判断する。「Ａ」以下であればステップＳ１６に進み、「Ａ」以下でなければステップＳ１７に進む。

［Ｓ１６］隣接する弟の辺節のアドレスが「Ａ」以下の場合、隣接する弟の辺節に注目し、ステップＳ１２に進む。

［Ｓ１７］現在の辺節が子を有しているか否かを判断する。子を有していればステップＳ１８に進み、子を有していなければ不正終了として処理を終了する。

［Ｓ１８］現在の辺節が子を有している場合には、長子の辺節のアドレスが「Ａ」以下であるか否かを判断する。「Ａ」以下であればステップＳ１９に進み、「Ａ」以下でなければ、不正終了として処理を終了する。

［Ｓ１９］現在のアークのラベルをラベル記録部にプッシュして、長子の辺節に注目し、ステップＳ１２に進む。

【００５２】以上の処理によって、ノード位置をキーとして対応する関連単語が出力される。例として、単語へのリンク情報としてアドレス「１４」が得られたとき、上記のアルゴリズムにしたがって図７のように記録されたトライから対応する関連単語を求めることにする。

【００５３】まず、ステップＳ１１より、「Ａ」に「１４」を代入し、アドレス「０」の辺節に注目する。ラベル記録部は空にしておく。以下、辺節はそのアドレスによって識別することにする。すなわち、アドレス「０」の辺節は、辺節「０」とよぶことにする。ノードとアークに関しても辺節と同様に識別することにする。

【００５４】ステップＳ１２において、現在の辺節「０」は「Ａ（＝１４）」に等しくないので、ステップＳ１４に進む。ステップＳ１４において、現在の辺節「０」には弟の辺節が存在するので、ステップＳ１５に進む。ステップＳ１５において、隣接する弟＝２４、Ａ＝１４であり、隣接する弟≦Ａは成り立たないので、ステップＳ１７に進む。

【００５５】ステップＳ１７において、現在の辺節「０」には子の辺節が存在するので、ステップＳ１８に進む。ステップＳ１８において、長子＝５、Ａ＝１４であり、長子≦Ａが成り立つので、ステップＳ１９に進む。

【００５６】ステップＳ１９において、現在のアーク「０」のラベル「解」をラベル記録部にプッシュし、長子の辺節「５」に注目する。そして、ステップＳ１２に進む。再びステップＳ１２において、現在の辺節「５」は「Ａ（＝１４）」に等しくないので、ステップＳ１４に進む。

【００５７】ステップＳ１４では、現在の辺節「５」に対して弟の辺節が存在するので、ステップＳ１５に進む。ステップＳ１５では、隣接する弟（＝１０）≦Ａ（＝１４）が成り立つので、隣接する弟「１０」に注目し、ステップＳ１２に進む。

【００５８】ステップＳ１２において、現在の辺節「１０」は「Ａ（＝１４）」と等しくないので、ステップＳ１４に進む。ステップＳ１４において、現在の辺節「１０」に対して弟の辺節が存在しないので、ステップＳ１７に進む。

【００５９】ステップＳ１７において、現在の辺節「１

０」には子の辺節が存在するので、ステップＳ１８に進む。ステップＳ１８において、長子＝１４、Ａ＝１４であり、長子≦Ａが成り立つので、ステップＳ１９に進む。

【００６０】ステップＳ１９において、現在のアーク「１０」のラベル「像」をラベル記録部にプッシュし、長子の辺節「１４」に注目する。そして、ステップＳ１２に進む。

【００６１】ステップＳ１２において、現在の辺節「１４」は「Ａ（＝１４）」と等しいのでステップＳ１３に進む。ステップＳ１３において、現在のアーク「１４」のラベル「度」をラベル記録部にプッシュし、ラベル記録部の内容である「解、像、度」を出力し、正常に処理を終える。

【００６２】以上の処理によって、ノードの位置「１４」から、対応する単語である「解像度」を得ることができる。同様にして、ノードの位置「１９」からは、「解像力」が得られる。すなわち、「解像度」の入力に対して、「解像度、解像力」の出力が得られたことになる。

【００６３】このようにして、トライを用いて、関連語の検索処理を少ない記憶容量で高速に行うことが可能となる。次に、第２の実施の形態について説明する。第２の実施の形態は、文字からその文字を含むキーワードを検索する単語検索装置である。

【００６４】図１０は、第２の実施の形態の概略構成を示す図である。この実施の形態に係る単語検索装置は、単語格納部５１、文字インデックス部５２、ノード位置検索部５３、及び単語検索部５４とから構成されている。この構成要素のうち、単語格納部５１、ノード位置検索部５３及び単語検索部５４は、図３に示した、単語格納手段１１、ノード位置検索手段１４及び単語検索手段１２の機能を有している。また、文字インデックス部５２は、図３のキーインデックス格納手段１３のキーインデックスを具体的な文字インデックスとしたものである。

【００６５】はじめに、索引単語の集合をトライで表現し、単語格納部５１を構成する。索引単語を表す最終ノードには、その索引単語に関連する文書の集合をたどれるようにリンクを張っておく。第１の実施の形態と同様に、図５のように与えられた索引単語に対応して図６のトライによる単語インデックス（トライ・インデックス）を構成する。この単語インデックスは、単語格納部５１に格納される。

【００６６】さらに、索引単語を構成する文字からその索引単語を導くことができる文字インデックスを構成する。この文字インデックスは、文字インデックス部５２に格納される。図５の索引単語集合の場合、索引単語を構成する全ての文字の集合は、｛解，現，析，像，度，力｝である。この文字集合の各文字について、図６のトライを先頭からたどることによって、各々の文字の位置を求める。その結果から、文字インデックスを作成する。

【００６７】図１１は、文字インデックスの例を示す図である。図のように、文字とその文字を含む索引単語を対応させる。この例では、トライ５５を文字インデックスのデータ構造として採用している。

【００６８】図５から、文字「解」を含む索引単語は「解」「解析」「解像度」「解像力」と複数あることが分かるが、これらへのリンク情報は「解」という文字を表す１つのノードの位置「０」だけで済んでいる。このように、トライ・インデックス中での文字のノードの位置を用いると、単語の末尾のノード位置を単語へのポインタとする場合に比べて、文字インデックスの容量を小さくすることができる。

【００６９】さて、検索を実行する動作を例を交えて説明する。検索キーとして１文字が入力され、この文字を含む索引単語と文書集合へのリンクとを求める場合を考えることにする。具体例として、図１１および図６のインデックスを用いて、検索キーとして「像」が入力された場合を想定する。

【００７０】まず、ノード位置検索部５３によって、文字インデックス部５２から検索キーの文字（例では「像」）を含んでいる索引単語へのリンクを見つける。図１１より、「１０」、「２８」、「３３」というリンク情報が得られることが分かる。

【００７１】次に、単語検索部５４によって、リンク情報から、単語インデックスを用いて索引単語およびその索引単語が指し示す文書集合を求める。以下に、その手順を説明する。

【００７２】図１２～図１４は、第２の実施の形態において文書集合を求めるための処理手順を示すフローチャートである。図１２ではステップＳ２１～Ｓ２７の処理を示しており、図１３ではステップＳ３１～Ｓ３６の処理を示しており、図１４ではステップＳ４１～Ｓ４８の処理を示している。以下、各ステップの処理内容を説明する。

［Ｓ２１］求める索引単語へのリンク情報は、トライの辺節アドレスが「Ａ」で与えられたとする。索引単語の文字列を記録するためのラベル記録部、ラベル記録部に記録した文字列を一時的に待避させておくためのラベルスタック、及びラベルを待避させたときの辺節を保存しておくための辺節スタックをそれぞれ用意し、内容をクリアしておく。開始ノードに移動し、トライの先頭にある辺節に注目する。注目している辺節を現在の辺節とよぶことにする。

［Ｓ２２］現在の辺節が、与えられたリンク情報「Ａ」と等しいか否かを判断する。等しければステップＳ２３に進み、等しくなければステップＳ３１（図１３に示す）へ進む。

15

［Ｓ２３］現在の辺節が、与えられたリンク情報「Ａ」と等しい場合、現在のアークのラベルをラベル記録部にプッシュする。

［Ｓ２４］現在のノードが終了状態か否かを判断する。終了状態であればステップＳ２５に進み、終了状態でなければステップＳ２６に進む。

［Ｓ２５］現在のノードが終了状態の場合、ラベル記録部の内容と、現在の終了状態ノードに対応する文書集合へのリンクとをそれぞれ出力する。

［Ｓ２６］現在の辺節が子を持っている否かを判断する。子をもっていればステップＳ２７に進み、子をもっていなければ、処理を正常終了する。

［Ｓ２７］現在の辺節が子をもっている場合、長子の辺節に注目し、ステップＳ４１（図１４に示す）に進む。

［Ｓ３１］ステップＳ２２において、現在の辺節が弟をもっていると判断された場合、ステップＳ３２に進み、弟をもっていなければステップＳ３４に進む。

［Ｓ３２］隣接する弟の辺節のアドレスが「Ａ」以下か否かを判断する。「Ａ」以下であればステップＳ３３に進み、「Ａ」以下でなければステップＳ３４に進む。

［Ｓ３３］隣接する弟の辺節のアドレスが「Ａ」以下の場合、隣接する弟の辺節に注目し、ステップＳ２２（図１２に示す）に進む。

［Ｓ３４］隣接する弟の辺節のアドレスが「Ａ」以下でない場合、現在の辺節が子をもっているか否かを判断する。子をもっていればステップＳ３５に進み、子をもっていなければ不正終了として処理を終了する。

［Ｓ３５］長子の辺節のアドレスが「Ａ」以下か否かを判断し、「Ａ」以下であればステップＳ３６に進み、「Ａ」以下でなければ不正終了として処理を終了する。

［Ｓ３６］現在のアークのラベルをラベル記録部にプッシュして、長子の辺節に注目する。そして、ステップＳ２２（図１２に示す）に進む。

［Ｓ４１］現在の辺節が弟をもっているか否かを判断する。弟をもっていればステップＳ４２に進み、弟をもっていなければステップＳ４３に進む。

［Ｓ４２］現在の辺節が弟をもっている場合には、ラベル記録部の内容をラベルスタックに、隣接する弟の辺節を辺節スタックにそれぞれプッシュする。

［Ｓ４３］現在のノードが終了状態か否かを判断する。終了状態であればステップＳ４４に進み、終了状態でなければステップＳ４５に進む。

［Ｓ４４］現在のノードが終了状態の場合、ラベル記録部の内容と現在のアークのラベルをつなげたもの、現在の終了状態ノードに対応する文書集合へのリンクとをそれぞれ出力する。

［Ｓ４５］現在の辺節が子をもっているか否かを判断する。子をもっていればステップＳ４６に進み、子をもっていなければステップＳ４７に進む。

［Ｓ４６］現在の辺節が子をもっている場合、現在のアークのラベルをラベル記録部にプッシュして、長子の辺節に注目する。そして、ステップＳ４１に進む。

16

［Ｓ４７］ラベルスタックと辺節スタックが空か否かを判断する。空でなければステップＳ４８に進み、空であれば正常終了する。

［Ｓ４８］ラベルスタックと辺節スタックが空ではないとき、ラベルスタックからラベル記録部にポップし、辺節スタックからポップした辺節に注目する。そして、ステップＳ４１に進む。

【００７３】以上の処理によって、索引単語の文字列と、その索引単語から対応する文書集合へのリンクが出力される。例として、索引単語へのリンク情報として「１０」が得られたとき、上記のアルゴリズムにしたがって図７のトライをたどることにする。

【００７４】まず、ステップＳ２１より、「Ａ」に「１０」を代入し、辺節「０」に注目する。ラベル記録部、ラベルスタック、辺節スタックはいずれも空にしておく。ステップＳ２２において、現在のノード「０」は「Ａ（＝１０）」に等しくないので、ステップＳ３１に進む。

【００７５】ステップＳ３１において、現在のノード「０」には弟が存在するので、ステップＳ３２に進む。ステップＳ３２において、隣接する弟＝２４、Ａ＝１０であり、隣接する弟≦Ａは成り立たないので、ステップ３４に進む。

【００７６】ステップＳ３４において、現在のノード「０」には子が存在するので、ステップＳ３５に進む。ステップＳ３５において、長子＝５、Ａ＝１０であり、長子≦Ａが成り立つので、現在のアーク「０」のラベル「解」をラベル記録部にプッシュし、長子の辺節「５」に注目する。そして、ステップＳ２２に進む。

【００７７】再びステップＳ２２において、現在の辺節「５」は「Ａ（＝１０）」に等しくないので、ステップＳ３１に移る。ステップＳ３１では、現在の辺節「５」に対して弟の辺節「１０」が存在するので、ステップＳ３２に進む。ステップＳ３２では、隣接する弟（＝１０）≦Ａ（＝１０）は成り立つので、隣接する弟「１０」に注目し、ステップＳ２２に進む。

【００７８】ステップＳ２２において、現在の辺節「１０」は「Ａ（＝１０）」と等しいので、現在のアーク「１０」のラベル「像」をラベル記録部にプッシュする。現時点のラベル記録部の内容は「解像」である。ステップＳ２４に進む。ステップＳ２４において、現在のノード「１０」は終了状態ではないので、ステップＳ２６に進む。ステップＳ２６において、現在の辺節「１０」は子をもっているので、長子の辺節「１４」に注目し、ステップＳ４１に進む。

【００７９】ステップＳ４１において、現在のノード「１４」は弟をもっているので、ステップＳ４２に進む。ステップＳ４２において、ラベル記録部の内容であ

る「解像」をラベルスタックにプッシュし、隣接する弟の辺節「１９」を辺節スタックにプッシュする。そして、ステップ４３に進む。

【００８０】ステップＳ４３において、現在のノード「１４」は終了状態なので、ステップＳ４４に進む。ステップＳ４４において、ラベル記憶部の内容である「解像」とアーク「１４」のラベルである「度」をつなげた「解像度」と、ノード「１４」に対応する文書集合へのリンク情報である「Ｔ３」を出力する。そして、ステップＳ４５に進む。

【００８１】ステップＳ４５において、現在のノード「１４」は子をもたないので、ステップＳ４７に進む。ステップＳ４７において、ラベルスタックと辺節スタックは空ではないので、ステップＳ４８に進む。ステップＳ４８において、ラベルスタックからポップして「解像」を取り出し、これをラベル記録部に代入し、辺節スタックからポップして辺節「１９」を取り出し、この辺節「１９」に注目する。現時点では、ラベルスタックと辺節スタックは空である。そして、ステップＳ４１に進む。

【００８２】再びステップＳ４１において、現在のノード「１９」は弟をもたないので、ステップＳ４３に進む。ステップＳ４３では、現在のノード「１９」は終了状態なので、ステップＳ４４に進む。ステップＳ４４において、ラベル記憶部の内容である「解像」とアーク「１９」のラベルである「力」をつなげた「解像力」と、ノード「１９」に対応する文書集合へのリンク情報である「Ｔ４」を出力する。そして、ステップＳ４５に進む。

【００８３】ステップＳ４５において、現在のノード「１９」は子をもたないので、ステップＳ４７に進む。ステップＳ４７において、ラベルスタックと辺節スタックは空なので、正常に処理を終わる。

【００８４】以上の処理によって、「像」という文字を含む索引単語へのリンクである「１０」から、２つの索引単語（「解像度」と「解像力」）、および対応する文書集合へのリンク情報（「Ｔ３」と「Ｔ４」）を求めることができる。

【００８５】次に第３の実施の形態について説明する。図１５は、第３の実施の形態の概略構成を示すブロック図である。これは、任意の正規表現からその正規表現を満たすキーワードを検索する単語検索装置である。

【００８６】この実施の形態に係る単語検索装置は、単語格納部６１、文字インデックス部６２、ノード位置検索部６３、単語検索部６４、及び正規表現解析部６５で構成されている。このうち、単語格納部６１、文字インデックス部６２、ノード位置検索部６３、及び単語検索部６４は、図１０に示した単語格納部５１、文字インデックス部５２、ノード位置検索部５３、及び単語検索部５４とほぼ同じ機能を有している。

【００８７】正規表現解析部６５は、正規表現の検索キーが入力されると、検索キーを解析し、その検索キーに適合する単語集合を得る。その際、必要に応じて、ノード位置検索部６３へ文字列を入力し、その戻り値として各文字列のノード位置を得る。また、ノード位置を単語検索部６４に入力して、単語集合を得る。

【００８８】以下に、正規表現解析部６５が検索を実行する際の動作を説明する。なお、トライにおいて、２つのノード「Ｎ１」、「Ｎ２」を指定したときに、ノード「Ｎ１」からたどることができ、ノード「Ｎ２」からはたどることのできないような、「Ｎ１」を開始ノードとするトライの部分木を、「Ｎ１」と「Ｎ２」から規定されるサブトライとよぶことにする。

【００８９】ここで、以下の３つの手続き関数「Ｆ１」、「Ｆ２」、「Ｆ３」を定義する。これらの関数の処理は、正規表現解析部６５で行われる。まず、関数「Ｆ１」について説明する。関数「Ｆ１」は、サブトライ「Ｔ」と正規表現「Ｒ」を引数とし、文字列の集合「Ｓ」を値として返す関数である。

【００９０】図１６、図１７は、関数「Ｆ１」の処理手順を示すフローチャートである。図１６は、ステップＳ５１～Ｓ５８の処理を示し、図１７は、ステップＳ６１～Ｓ６８の処理を示している。

［Ｓ５１］正規表現「Ｒ」の先頭が、確定している文字列「Ｂ」または文字集合「Ｂ」で始まるか否かを判断する。確定文字列等で始まる場合にはステップＳ５２に進み、確定文字列で始まらない場合にはステップＳ６１に進む。

［Ｓ５２］正規表現「Ｒ」の先頭が確定している文字列「Ｂ」または文字集合「Ｂ」から始まる場合、「Ｒ」から「Ｂ」の部分を除く正規表現を「Ｒ１」とし、以下の処理を行う。

［Ｓ５３］文字列または文字集合「Ｂ」を入力としてサブトライ「Ｔ」をたどることができるか否かを判断する。たどることができた場合には、ステップＳ５５に進み、たどることができない場合にはステップＳ５４に進む。

［Ｓ５４］エラーを関数「Ｆ１」の値「Ｓ」として返し、処理を終了する。

［Ｓ５５］「Ｂ」から「Ｔ」をたどることができた場合、正規表現「Ｒ１」が空か否かを判断する。空であればステップＳ５６に進み、空でなければステップＳ５８へ進む。

［Ｓ５６］正規表現「Ｒ１」が空の場合、サブトライ「Ｔ」を入力「Ｂ」でたどった先が終了状態のノードか否かを判断する。終了状態であればステップＳ５７に進み、終了状態でなければステップＳ５４に進む。

［Ｓ５７］サブトライ「Ｔ」を入力「Ｂ」でたどった先が終了状態のノードの場合、文字列「Ｂ」を関数「Ｆ１」の値「Ｓ」として返し、処理を終了する。

［Ｓ５８］正規表現「Ｒ１」が空でない場合、文字列「Ｂ」を入力としてサブトライ「Ｔ」をたどったとき、更にたどることのできる残りのサブトライを「Ｔ１」とする。関数「Ｆ１」に、引数としてサブトライ「Ｔ１」と正規表現「Ｒ１」を渡し、文字列「Ｂ」と関数「Ｆ１」の評価した値である文字列集合の各々とを連結し、得られた文字列集合を値「Ｓ」として返す。そして、処理を終了する。

［Ｓ６１］正規表現「Ｒ」の中に、確定している文字「Ｃ１，Ｃ２，Ｃ３，・・・」が含まれているか否かを判断する。確定している文字が含まれていればステップＳ６２に進み、含まれていなければステップＳ６８に進む。

［Ｓ６２］確定している文字Ｃｉ（ただし、ｉ＝１，２，３，・・・）をトライにおいて表しているノードが出現する数をＡｉとし、Ａ１，Ａ２，Ａ３，... の中で最小値をとる文字「Ｃｊ」を、ノード位置検索部６３の出力から決定する。

［Ｓ６３］文字「Ｃｊ」を表しているノード位置の各々「Ｐｉ（ただし、ｉ＝１，２，３，・・・）」について、正規表現「Ｒ」において文字「Ｃｊ」を最後に含む「Ｒ」の一部を正規表現「Ｒ１」とし、残りの正規表現を「Ｒ２」とし、「Ｒ１」を受理する有限状態オートマトン「Ｍ」をつくり、ステップＳ６４に進む。すべてのノード位置「Ｐｉ」について処理を終えたとき、ステップＳ６７に進む。

［Ｓ６４］関数「Ｆ２」に、引数として有限状態オートマトン「Ｍ」、サブトライ「Ｔｉ」、ノード位置「Ｐｉ」を渡し、関数「Ｆ２」を評価した値である文字列「ｓ」を得る。

［Ｓ６５］関数「Ｆ２」の値である文字列「ｓ」が正常出力か否かを判断する。正常出力であればステップＳ６６に進み、正常出力でなければ（文字列「ｓ」がエラーの場合）ステップＳ６３に進む。

［Ｓ６６］ノード位置「Ｐｉ」からたどることのできるサブトライを「Ｔｉ」とする。関数「Ｆ１」に、引数としてサブトライ「Ｔｉ」と正規表現「Ｒ２」を渡し、文字列「ｓ」と、関数「Ｆ１」を評価した値である文字列集合「Ｓ１」の各々の要素を連結した文字列の集合を求め、文字列集合「Ｓ」としてプッシュする。そして、ステップＳ６３へ進む。

［Ｓ６７］文字列集合「Ｓ」を出力し、処理を終える。

［Ｓ６８］「Ｒ」を満たす文字数「Ｎ」を求め、関数「Ｆ３」に、引数としてサブトライ「Ｔ」と文字数「Ｎ」を渡し、関数「Ｆ３」を評価した値である文字列集合を求め、文字列集合「Ｓ」として出力し、処理を終える。

【００９１】次に、関数「Ｆ２」について説明する。関数「Ｆ２」は、有限状態オートマトン「Ｍ」とサブトライ「Ｔ」、ノード位置「Ｐ」を引数とし、文字列「ｓ」を値として返す関数である。

【００９２】図１８は、関数Ｆ２の処理手順を示すフローチャートである。

［Ｓ７１］サブトライ「Ｔ」において、ノード位置「Ｐ」を指定すると、そのノードに至るまでの経路は一意に定まる。そこで、サブトライ「Ｔ」をノード位置「Ｐ」に至るまでたどることによって得られるラベル列「Ｌ」を、有限状態オートマトン「Ｍ」の入力とする。

［Ｓ７２］ラベル列「Ｌ」が有限状態オートマトン「Ｍ」に受理されるか否かを判断する。受理される場合はステップＳ７３に進み、受理されない場合はステップＳ７４に進む。

［Ｓ７３］ラベル列「Ｌ」が有限状態オートマトン「Ｍ」に受理される場合、ラベル列「Ｌ」を出力し、処理を終了する。

［Ｓ７４］ラベル列「Ｌ」が有限状態オートマトン「Ｍ」に受理されない場合、エラーを出力し、処理を終了する。

【００９３】次に、関数「Ｆ３」について説明する。関数「Ｆ３」は、サブトライ「Ｔ」と文字数「Ｎ」を引数とし、文字列の集合「Ｓ」を値として返す。図１９は、関数Ｆ３の処理手順を示すフローチャートである。

［Ｓ８１］開始ノードが終了状態であり、かつ文字数「Ｎ」として「０」をとることができる場合、空の文字列を文字列の集合「Ｓ」にプッシュする。

［Ｓ８２］サブトライ「Ｔ」を文字数「Ｎ」分だけすべてたどり、たどった先が終了状態のノードとなるラベル列の集合「Ｌ」を求め、「Ｌ」を文字列の集合「Ｓ」にプッシュする。

［Ｓ８３］「Ｓ」が空か否かを判断する。「Ｓ」が空であればステップＳ８５に進み、「Ｓ」が空でないならステップＳ８４に進む。

［Ｓ８４］「Ｓ」が空でない場合、「Ｓ」を出力し、処理を終了する。

［Ｓ８５］「Ｓ」が空の場合、エラーを出力し、処理を終了する。

【００９４】以上のように関数を定義すると、関数「Ｆ１」に、引数として単語インデックスのトライと任意の正規表現を渡し、関数「Ｆ１」を評価することによって、与えた正規表現に適合する単語の集合を得ることができる。

【００９５】具体例として、正規表現「？アイデ？ア？」に合致する単語を探す場合を考えることにする。正規表現において、「？」は０個以上の任意の文字に該当するワイルドカードを意味する。検索の意図は、「アイデア」に対して「アイディア」などといった表記の揺れを考慮し、かつ「アイデ？ア」という文字列を含む単語を検索することである。このように、正規表現に合致する単語を検索できることによって、検索洩れを少なくすることが期待できる。

【００９６】例えば、以下のような単語インデックスが単語格納部６１に格納されている場合を考える。図２０は、第３の実施の形態におけるトライ６６の例を示す図である。この例では、各辺節のラベルとして、カタカナ１文字が与えられている。

【００９７】図２１は、第３の実施の形態における文字インデックスの例を示す図である。この文字インデックスでは、各文字に対して、「出現数」と「対応するノード位置」とが対応付けられている。

【００９８】ここで、関数「Ｆ１」に、引数として単語インデックスのトライ「Ｔ」と正規表現「？アイデ？ア？」（Ｒ）を渡した場合の、関数「Ｆ１」の値の評価手順を説明する。

【００９９】ステップＳ５１において、正規表現「Ｒ」は確定文字列から始まっていなので、ステップＳ６１へ進む。ステップＳ６１において、正規表現「Ｒ」の中に、確定している文字（「ア」「イ」「デ」）が含まれているので、ステップＳ６２へ進む。

【０１００】ステップＳ６２において、「Ｒ」の確定している３つの文字｛Ｃ１，Ｃ２，Ｃ３｝＝｛ア，イ，デ｝を表すノードの出現数をそれぞれ求める。図２１の文字インデックスからそれぞれ、「５」「２」「２」であると分かる。

【０１０１】ステップＳ６３において、これらの中で最小値をとる文字として「イ」をとりあげる。そして、文字「イ」を表しているすべてのノード位置｛Ｐ１，Ｐ２｝＝｛１０，１２０｝について、以下の処理を行う。

【０１０２】ステップＳ６３で、正規表現「？アイデ？ア？」から、「Ｒ１」を「？アイ」、「Ｒ２」を「デ？ア？」とし、「Ｒ１」を受理する有限オートマトンを「Ｍ」を作る。

【０１０３】図２２は、「イ」をとりあげた場合の有限オートマトン６７の遷移図である。図に示すように、文字列の途中（最初でもよい）で「ア」、「イ」が連続して出現した場合に、終了状態のノードへ遷移する。

【０１０４】ステップＳ６４で、関数「Ｆ２」に、引数として有限状態オートマトン「Ｍ」、サブトライ「Ｔ」、ノード位置「Ｐ１」を渡し、関数「Ｆ２」を評価する。さて、関数「Ｆ２」の処理（図１８に示す）のステップＳ７１において、サブトライ「Ｔ１」をノード位置「Ｐ１（＝１０）」に至るまでに得られるラベル列「Ｌ」は「アイ」である。

【０１０５】ステップＳ７２の判断において、「Ｌ」は有限状態オートマトン「Ｍ」に受理されることが分かる。ステップＳ７３で、ラベル列「Ｌ」（＝「アイ」）を出力し、関数「Ｆ２」の処理を完了する。

【０１０６】関数「Ｆ２」の値は、文字列「アイ」である。したがって、ステップＳ６４では、これを「ｓ」とする。ステップＳ６５で、文字列「ｓ」（＝「アイ」）はエラーではないので、ステップＳ６６へ進む。

【０１０７】ステップＳ６６で、関数「Ｆ１」に、引数としてサブトライ「Ｔ１」と正規表現「Ｒ２」（＝「デ？ア？」）を渡し、関数「Ｆ１」を評価する。以後、この関数「Ｆ１」に関しては関数「Ｆ１’」などと表記する。

【０１０８】図２３は、ノード位置Ｐ１（＝１０）からたどることのできるサブトライ「Ｔ１」を示す図である。このサブトライ６８は、図２０のトライ６６におけるノード位置Ｐ１（＝１０）の子孫に該当する全ての経路を抽出したものである。

【０１０９】さて、関数「Ｆ１’」のステップＳ５１において、正規表現「Ｒ’」の先頭は確定している文字「Ｂ’」（＝「デ」）から始まっている。ステップＳ５２において、「Ｒ１’」を「？ア？」とし、ステップＳ５３へ進む。

【０１１０】関数「Ｆ１’」のステップＳ５３で、文字「Ｂ’」（＝「デ」）を入力としてサブトライ「Ｔ’」をたどることができるので、ステップＳ５５へ進む。関数「Ｆ１’」のステップＳ５５の判断において、正規表現「Ｒ１’」「？ア？」は空ではないので、ステップＳ５８へ進む。

【０１１１】関数「Ｆ１’」のステップＳ５８で、文字「Ｂ’」（＝「デ」）を入力としてサブトライ「Ｔ’」をたどり、更にたどることのできる残りのサブトライを「Ｔ１’」とする。関数「Ｆ１」に、引数としてサブトライ「Ｔ１’」と正規表現「Ｒ１’」（＝「？ア？」）を渡し、関数「Ｆ１」を評価する。以後、このＦ１に関してはＦ１’’などと表記する。

【０１１２】さて、関数「Ｆ１’’」のステップＳ５１において、正規表現「Ｒ’’」（＝「？ア？」）の先頭は確定文字列から始まっていないので、ステップＳ６１へ進む。

【０１１３】関数「Ｆ１’’」のステップＳ５１において、正規表現「Ｒ’’」（＝「？ア？」）の中に確定文字「Ｃ１’’」（＝「ア」）が含まれているので、ステップＳ６２へ進む。

【０１１４】関数「Ｆ１’’」のステップＳ６２で、「Ｃｊ’’」として「Ｃ１’’」（＝「ア」）が相当し、ノード位置の集合｛Ｐ１’’，Ｐ２’’｝＝｛３０，８０｝が得られる。

【０１１５】「Ｃ１’’」（＝「ア」）について、関数「Ｆ１’’」のステップＳ６３以下の処理を行う。関数「Ｆ１’’」のステップＳ６３で、「Ｒ１’’」として「？ア」、「Ｒ２’’」として「？」をとる。「Ｒ１’’」を受理する有限状態オートマトン「Ｍ’’」をつくる。

【０１１６】関数「Ｆ１’’」のステップＳ６４で、関数「Ｆ２」に、引数として有限状態オートマトン「Ｍ’’」、サブトライ「Ｔ’’」、ノード位置「Ｐ１’’」（＝３０）を渡し、関数「Ｆ２」を評価する。

以後、この関数「Ｆ２」に関して関数「Ｆ２’’」などと表記する。

【０１１７】さて、関数「Ｆ２’’」のステップＳ７１において、サブトライ「Ｔ’’」をノード位置「Ｐ１’’」（＝３０）に至るまでに得られるラベル列「Ｌ’’」を「ア」として、有限状態オートマトン「Ｍ’’」に入力する。

【０１１８】ステップＳ７２において、「Ｌ’’」（＝「ア」）は有限状態オートマトン「Ｍ’’」に受理されることが分かる。関数「Ｆ２’’」のステップＳ７３で、ラベル列「Ｌ’’」（＝「ア」）を出力し、関数「Ｆ２’’」の処理を完了する。

【０１１９】関数「Ｆ２’’」の値「ア」は、文字列である。関数「Ｆ１’’」のステップＳ６４では、これを「Ｓ’’」とする。関数「Ｆ１’’」のステップＳ６５で、文字列「Ｓ’’」（＝「ア」）はエラーではないので、ステップＳ６６へ進む。

【０１２０】関数「Ｆ１’’」のステップＳ６６で、ノード位置「Ｐ１’’」（＝３０）からたどることのサブトライを「Ｔ１’’」とし、関数「Ｆ１」に、引数としてサブトライ「Ｔ１’’」と正規表現「Ｒ２’’」（＝「？）を渡し、関数「Ｆ１」を評価する。以後、この関数「Ｆ１」に関しては関数「Ｆ１’’’」などと表記する。

【０１２１】さて、関数「Ｆ１’’’」のステップＳ５１において、正規表現「Ｒ’’’」（＝「？」）の先頭は確定文字列から始まっていないので、ステップＳ６１へ進む。

【０１２２】関数「Ｆ１’’’」のステップＳ６１で、正規表現「Ｒ’’’」（＝「？」）の中に確定文字は含まれていないので、ステップＳ６８へ進む。関数「Ｆ１’’’」のステップＳ６８で、「Ｒ’’’」（＝「？」）を満たす文字数「Ｎ’’’」は無制限である。関数「Ｆ３」に、引数としてサブトライ「Ｔ’’’」と文字数「Ｎ’’’」を渡し、関数「Ｆ３」を評価した値である文字列集合は｛「」，「リズム」｝であり、関数「Ｆ１’’’」の値として返す。

【０１２３】文字列「ｓ’’」（＝「ア」）と、関数「Ｆ１’’’」を評価した値である文字列集合｛「」，「リズム」｝の各々の要素を連結した文字列の集合は｛「ア」，「アリズム」｝である。関数「Ｆ１’’」のステップＳ６６で、これを文字列集合「Ｓ’’」にプッシュする。現時点での「Ｓ’’」の内容は、｛「ア」，「アリズム」｝である。そして、ステップＳ６３へ進む。

【０１２４】関数「Ｆ１’’」のステップＳ６３以後の処理で、ノード位置「Ｐ１」（＝３０）についての処理を終えたことになる。次に同様に「Ｐ２’’」についての処理を行い、再びステップＳ６３に戻ってきて、結果として、「Ｓ’’」＝｛「ア」，「アリズム」，「イア」｝を出力する。

【０１２５】関数「Ｆ１’」のステップＳ５８に戻り、文字「Ｂ’」（＝「デ」）と関数「Ｆ１’’」の値｛「ア」，「アリズム」，「ィア」｝の各々の要素とを連結し、関数「Ｆ１’」の値として｛「デア」，「デアリズム」，「ディア」｝を返す。

【０１２６】関数「Ｆ１」のステップＳ６６に戻り、文字列「ｓ」（＝「アイ」）と関数「Ｆ１’」の値｛「デア」，「デアリズム」，「ディア」｝の各々の要素とを連結し、文字列集合「Ｓ」に｛「アイデア」，「アイデアリズム」，「アイディア」｝をプッシュする。現時点での「Ｓ」の内容は、｛「アイデア」，「アイデアリズム」，「アイディア」｝である。ステップＳ６３へ進む。

【０１２７】関数「Ｆ１」のステップＳ６３において、文字「イ」を表しているノード位置「Ｐ２」（＝１２０）について以下の処理を行う。関数「Ｆ１」のステップＳ６４で、関数「Ｆ２」に、引数として有限状態オートマトン「Ｍ」、サブトライ「Ｔ」、ノード位置「Ｐ２」を渡し、関数「Ｆ２」を評価した値である文字列「ｓ」（＝「ネオアイ」）を得る。

【０１２８】関数「Ｆ１」のステップＳ６５で、文字列「ｓ」（＝「ネオアイ」）はエラーはないので、ステップＳ６６に進む。Ｆ１のステップＳ６６で、「Ｐ２」（＝１２０）からたどることのできるサブトライを「Ｔ２」とし、関数「Ｆ１」に、引数としてサブトライ「Ｔ２」と正規表現「Ｒ２」（＝「デ？ア？」）を渡し、関数「Ｆ１」を評価する。関数「Ｆ１」から文字列集合｛「デア」，「デアリズム」｝が得られ、「Ｓ」に｛「ネオアイデア」，「ネオアイデアリズム」｝をプッシュし、ステップＳ６３へ進む。現時点での「Ｓ」の内容は、｛「アイデア」，「アイデアリズム」，「アイディア」，「ネオアイデア」，「ネオアイデアリズム」｝である。

【０１２９】関数「Ｆ１」のステップＳ６３で、すべてのノード位置「Ｐ１」，「Ｐ２」について処理を終えたので、ステップＳ６７で、文字列集合「Ｓ」（＝｛「アイデア」，「アイデアリズム」，「アイディア」，「ネオアイデア」，「ネオアイデアリズム」｝）を関数「Ｆ１」の値として返し、処理を終える。

【０１３０】以上のように、単語インデックスのトライと正規表現「？アイデ？ア？」から、単語集合｛「アイデア」，「アイデアリズム」，「アイディア」，「ネオアイデア」，「ネオアイデアリズム」｝が求まる。

【０１３１】なお、上記の原理構成若しくは実施の形態は、以下のような変形例が考えられる。図２４は、第４の実施の形態の概略構成を示すブロック図である。これは、第１の実施の形態（図４に示す）における関連単語インデックス部２１を複数設けたものである。

【０１３２】この実施の形態では、２つの関連単語イン

デックス部７１，７２のそれぞれに、関連単語インデックスが格納されている。ノード位置検索部７３は、単語が入力されると、双方の関連単語インデックス部７１，７２からノードの位置集合を取得する。そのノードの位置の集合は、どちらの関連単語インデックス部７１，７２から取得したのかを示す情報と共に、単語検索部７４に渡される。

【０１３３】単語検索部７４は、ノード位置検索部７３から受け取ったノード位置の集合に基づいて、関連単語インデックス部７１，７２から関連単語の集合を取得し、出力する。

【０１３４】図２５は、第５の実施の形態の概略構成を示すブロック図である。これは、第４の実施の形態（図２４に示す）を具体化したものである。この実施の形態では、よみインデックス部７１ａと表記インデックス部７２ａが設けられている。よみインデックス部７１ａは、表記によって表された表記単語の集合を、深さ優先順にノードが記録されるトライ形式で格納していると共に、表記単語と各表記単語に対応する文字列を構成しているよみ単語のノードの位置の集合とを対応付けて格納している。表記インデックス部７２ａは、よみによって表された単語の集合を、深さ優先順にノードが記録されるトライ形式で格納していると共に、よみ単語と各よみ単語に対応する表記単語の文字列を構成しているノードの位置の集合とを対応付けて格納している。

【０１３５】ノード位置検索部７３ａと単語検索部７４ａとは、第４の実施の形態のノード位置検索部７３と単語検索部７４と同様の機能を有してる。図２６は、第６の実施の形態の概略構成を示すブロック図である。これは、第４の実施の形態の関連語インデックス部７１，７２をさらに増やしたものである。

【０１３６】この実施の形態では、多数の関連単語インデックス部８１ａ，８１ｂ，８１ｃ，・・・が設けられている。ノード位置検索部８２は、単語の入力を受け取ると、各関連単語インデックス部８１ａ，８１ｂ，８１ｃ，・・・から、該当するノードの位置の集合を受け取る。単語検索部８３は、ノード位置検索部８２から受け取ったノード位置の集合に基づいて、関連単語インデックス部８１ａ，８１ｂ，８１ｃ，・・・から関連単語の集合を取得し、出力する。

【０１３７】以上のように、本発明においては、単語の集合を、深さ優先順にノードが記録されるトライ形式にしたがって格納し、単語格納部を構成すると共に、トライにおいて単語を構成しているノードの位置をトライ中の単語を一意に識別できる値として用いることによって、親のノードへのリンク情報を用いることなく、任意のノードを含む経路を特定することができる。また、トライにおける単語を、トライにおけるノードの位置をポインタとして参照することによって、トライ・インデックス以外に別個に単語集合を表すデータは不要になる。

その結果、必要な記憶容量は、従来技術の場合に比べて、著しく少なくて済む。

【０１３８】なお、上記の処理機能は、コンピュータによって実現することができる。その場合、システム構築支援装置が有するべき機能の処理内容は、コンピュータで読み取り可能な記録媒体に記録されたプログラムに記述されており、このプログラムをコンピュータで実行することにより、上記処理がコンピュータで実現される。コンピュータで読み取り可能な記録媒体としては、磁気記録装置や半導体メモリ等がある。市場を流通させる場合には、ＣＤ－ＲＯＭ(Compact Disc Read Only Memory) やフロッピーディスケット等の可搬型記録媒体にプログラムを格納して流通させたり、ネットワークを介して接続されたコンピュータの記憶装置に格納しておき、ネットワークを通じて他のコンピュータに転送することもできる。コンピュータで実行する際には、コンピュータ内のハードディスク装置等にプログラムを格納しておき、メインメモリにロードして実行する。

【０１３９】

【実施例】本発明の実施例として、必要な記憶容量を、第５の実施の形態による場合と、従来技術の場合とを定量的に比較することにする。

【０１４０】図２７は、表記の単語とそれに対応するよみの単語の集合との対応関係を示す図である。図中左側に「表記で表される単語」が示されており、右側に「対応するよみ」が示されている。例えば、表記が「Ａ」の場合、「あるふぁ」とよむ場合もあれば、「えー」とよむ場合もある。

【０１４１】図２８は、よみの単語とそれに対応する表記の単語の集合との対応関係を示す図である。図中左側に「よみで表される単語」が示されており、右側に「対応する表記」が示されている。例えば、よみが「あ」の場合、その表記は、「あ」「ア」「亜」「阿」「在」「有」など多数ある。

【０１４２】図２７、図２８には、先頭の８語について示しているが、全体では、表記単語は９３，４５２語、よみ単語は６８，８１９語から成る。図２９は、第５の実施の形態におけるインデックス部の情報量を示す図である。このように、本発明を用いて、表記単語とよみ単語の対応データから、表記単語からよみ単語集合へのインデックスおよびよみ単語から表記単語へのインデックスをそれぞれ作成した結果、トライは１，２５７，５７９．０バイト、ポインタ・テーブルは４９４，０８５．０バイト、インデックス全体は１，７５１，６６４．０バイトの記憶容量となった。

【０１４３】そこで、従来技術の説明における第１の従来例、すなわち、単語の集合を固定長または可変長の文字列として格納するレコード構造のデータをインデックスとは別個に用意する方法で、同様の機能を果すための情報を格納した。

【０１４４】図３０は、第１の従来例における情報量を示す図である。第１の従来例では、テキストとその参照テーブルは２，０４２，５７０．５バイト、トライは１，２５７，５７９．０バイト、ポインタ・テーブルは４９４，０８５．０バイト、インデックス全体は３，７９４，２３４．５バイトの記憶容量となった。

【０１４５】また、従来技術の説明における第２の従来例、すなわち、トライ・インデックスを単語集合のデータとみなして、単語の末尾に対応するトライ中のノードの識別番号を単語へのポインタとする方法で，同様の機能を果すための情報を格納した。

【０１４６】図３１は、第２の従来例における情報量を示す図である。第２の従来例では、トライ・インデックスに、２，４６９，９９６．５バイト程度、ポインタ・テーブルは４９４，０８５．０バイト、そしてインデックス全体として２，９６４，０８１．５バイトの記憶容量を必要とすると予想できる。予想において、おおよそのトライデータのサイズから親ノードへのリンクに必要なデータ幅は２．５バイトと仮定し、トライのノードの数Ｎとし、親ノードへのリンクに必要なデータ容量Ｌを、Ｌ＝Ｎ×２．５（バイト）と計算した。

【０１４７】以上の結果に基づいて、第５の実施の形態と従来例とを比較した。図３２は、第５の実施の形態と従来技術との情報量の比較結果を示す図である。

【０１４８】この比較結果から、本発明の第５の実施の形態は、第１の従来例に比べてトライを除くインデックスは１９．５%、全体のインデックスは４６．２%、第２の従来例に比べてトライは５０．９%、全体のインデックスは５９．１%の記憶容量しか必要としないことが分かる。したがって、本発明によって、必要な記憶容量の著しい削減効果が得られたと言える。

【０１４９】

【発明の効果】以上説明したように本発明では、単語の集合を、深さ優先順にノードが記録されるトライ形式にしたがって格納し、トライにおいて単語を構成しているノードの位置をトライ中の単語を一意に識別できる値として用いたため、親のノードへのリンク情報を用いることなく、任意のノードを含む経路を特定することができる。また、トライにおけるノードの位置をポインタとして、トライにおける単語を参照するようにしたため、トライ形式の単語集合とは別個に単語集合を表すデータを用意する必要がない。その結果、必要な記憶容量は、従来技術の場合に比べて、著しく少なくて済む。

【０１５０】また、本発明の単語検索プログラムを記録した媒体では、記録されている単語検索プログラムをコンピュータで実行することにより、ノードの位置からそのノードに対応する単語を含む単語集合の検索を行う単語検索処理を、コンピュータの少ない記憶容量を使用して実行することが可能となる。

【図面の簡単な説明】

【図１】本発明の原理構成図である。

【図２】ノードの位置からそのノードを含む単語または単語の集合を求めるアルゴリズムを示すフローチャートである。

【図３】単語へのキーを入力とする単語検索装置の原理構成図である。

【図４】本発明の第１の実施の形態を示すブロック図である。

【図５】単語の集合の例を示す図である。

【図６】深さ優先順にノードが記録されたトライの例を示す図である。

【図７】トライ・インデックスの例を示す図である。

【図８】関連語対応テーブルを示す図である。

【図９】第１の実施の形態における単語検索手順を示すフローチャートである。

【図１０】第２の実施の形態の概略構成を示す図である。

【図１１】文字インデックスの例を示す図である。

【図１２】第２の実施の形態において文書集合を求めるための処理手順を示すフローチャート（その１）である。

【図１３】第２の実施の形態において文書集合を求めるための処理手順を示すフローチャート（その２）である。

【図１４】第２の実施の形態において文書集合を求めるための処理手順を示すフローチャート（その３）である。

【図１５】第３の実施の形態の概略構成を示すブロック図である。

【図１６】関数Ｆ１の処理手順を示すフローチャート（その１）である。

【図１７】関数Ｆ１の処理手順を示すフローチャート（その２）である。

【図１８】関数Ｆ２の処理手順を示すフローチャートである。

【図１９】関数Ｆ３の処理手順を示すフローチャートである。

【図２０】第３の実施の形態におけるトライの例を示す図である。

【図２１】第３の実施の形態における文字インデックスの例を示す図である。

【図２２】「イ」をとりあげた場合の有限オートマトンの遷移図である。

【図２３】ノード位置Ｐ１（＝１０）からたどることのできるサブトライ「Ｔ１」を示す図である。

【図２４】第４の実施の形態の概略構成を示すブロック図である。

【図２５】第５の実施の形態の概略構成を示すブロック図である。

【図２６】第６の実施の形態の概略構成を示すブロック

図である。

【図２７】表記の単語とそれに対応するよみの単語の集合との対応関係を示す図である。

【図２８】よみの単語とそれに対応する表記の単語の集合との対応関係を示す図である。

【図２９】第５の実施の形態におけるインデックス部の情報量を示す図である。

【図３０】第１の従来例における情報量を示す図である。

【図３１】第２の従来例における情報量を示す図である。

【図３２】第５の実施の形態と従来技術との情報量の比較結果を示す図である。

【符号の説明】

１　単語格納手段

２　単語検索手段

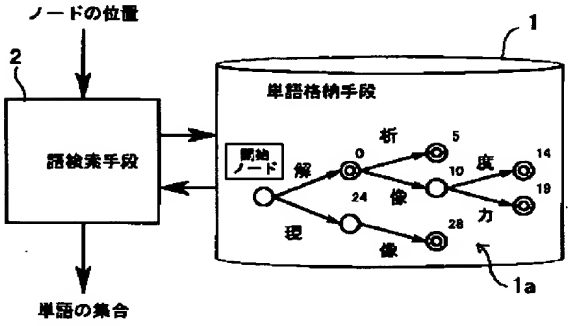
１１　単語格納手段

１２　キーインデックス格納手段

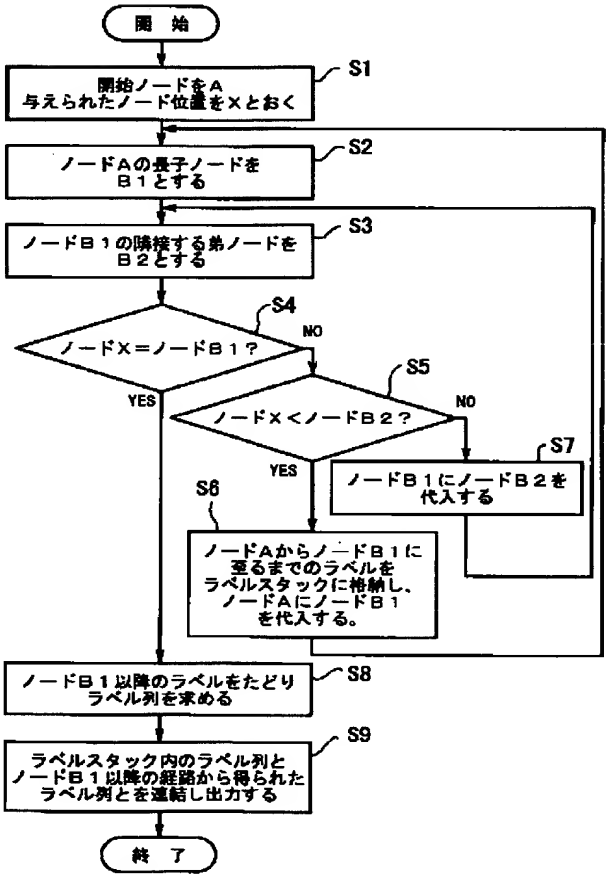
１３　ノード位置検索手段

１４　単語検索手段

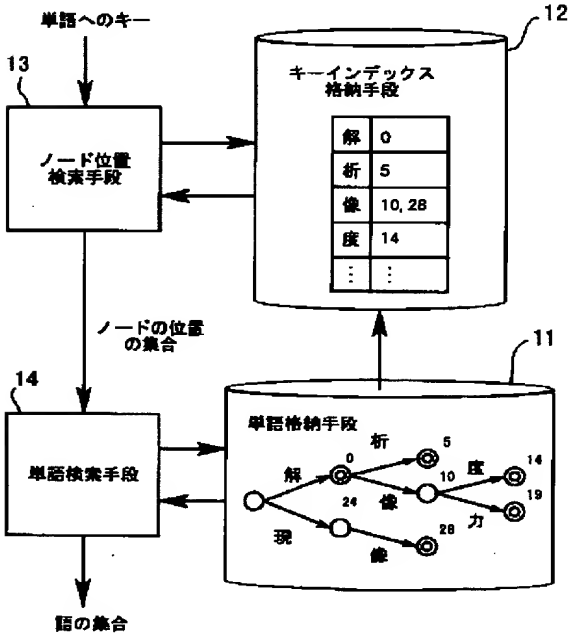
【図１】



【図２】



【図３】



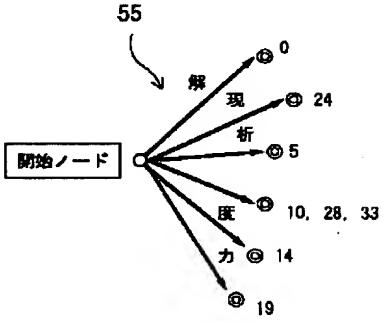
【図５】

単語	単語に対応するポインタ
解	T1
解析	T2
解像度	T3
解像力	T4
現像	T5
像	T6

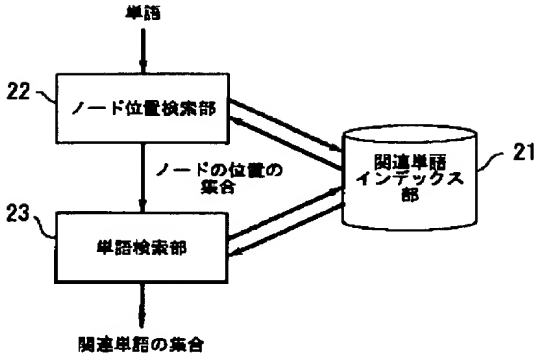
【図８】

単語からの関連単語集合へのリンク情報	ノード位置の集合
T3	14, 19
T4	14, 19

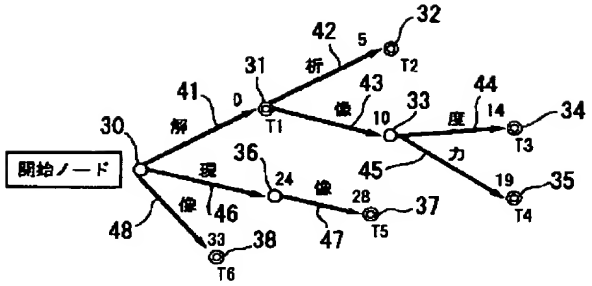
【図１１】



【図４】



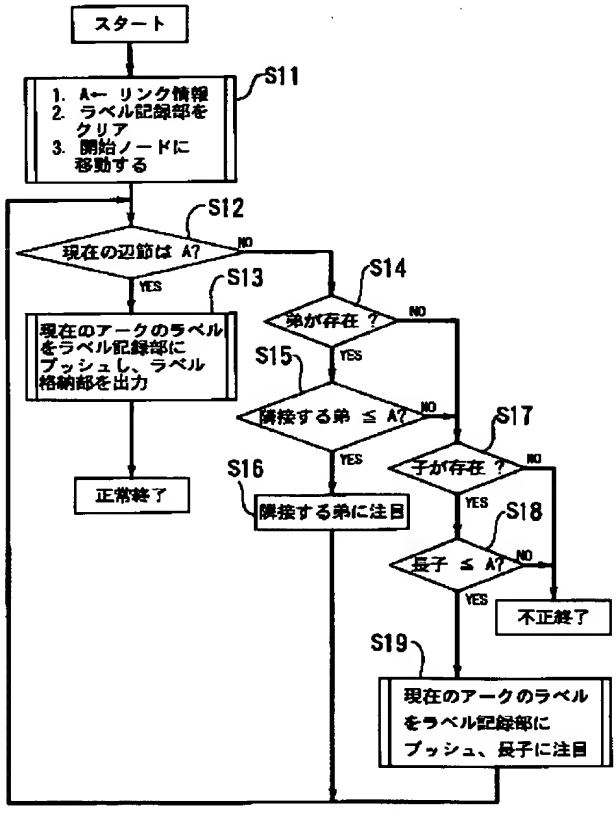
【図６】



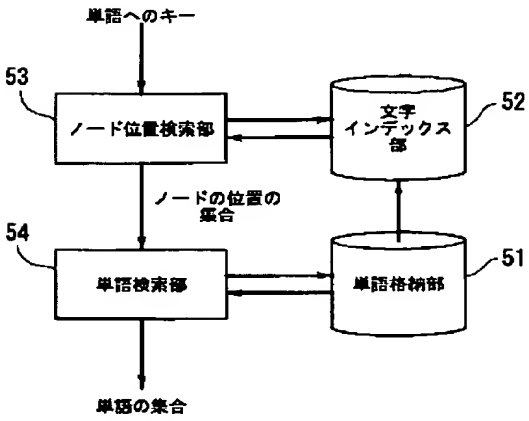
【図７】

アドレス	直下の弟ノードの位置	ラベル	ノードの状態	リンク情報
0	24	解	(終了・継続)	T1
5	10	析	(終了)	T2
10	-	像	(継続)	-
14	19	度	(終了)	T3
19	-	力	(終了)	T4
24	33	現	(継続)	-
28	-	像	(終了)	T5
33	-	像	(終了)	T6

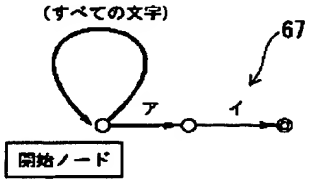
【図９】



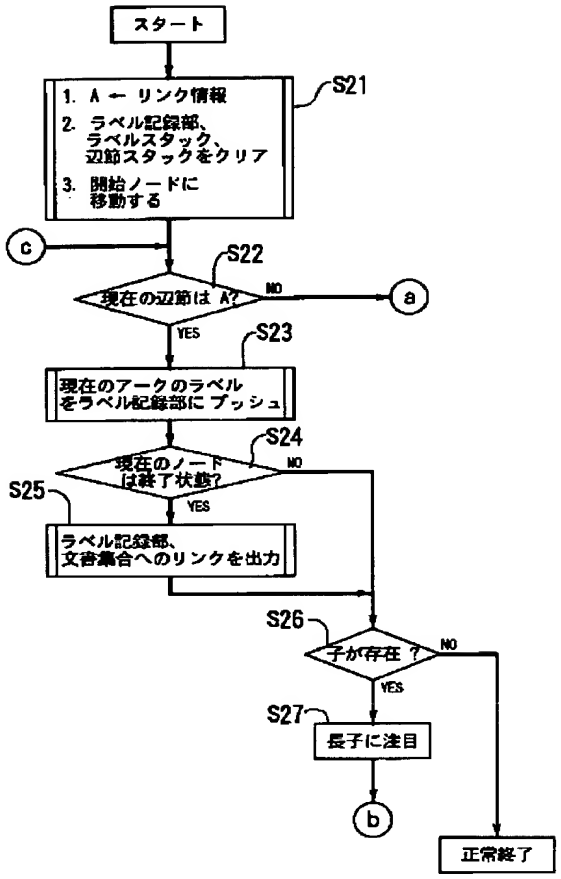
【図１０】



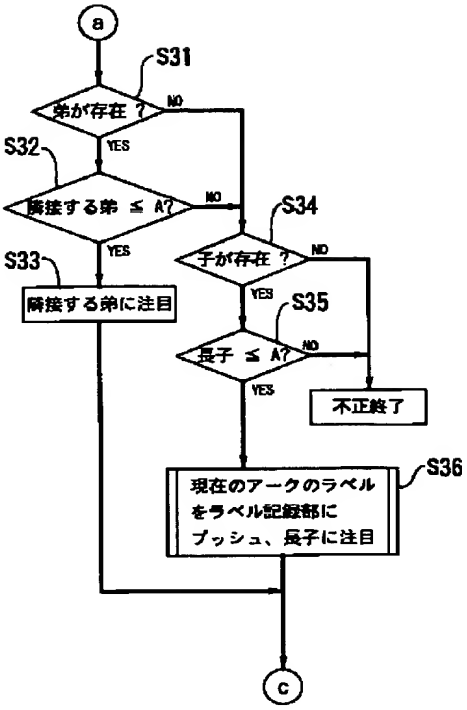
【図２２】



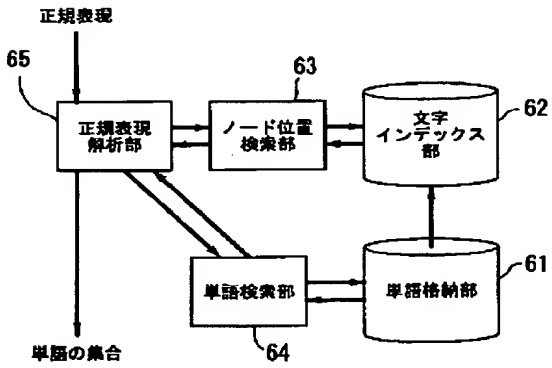
【図１２】



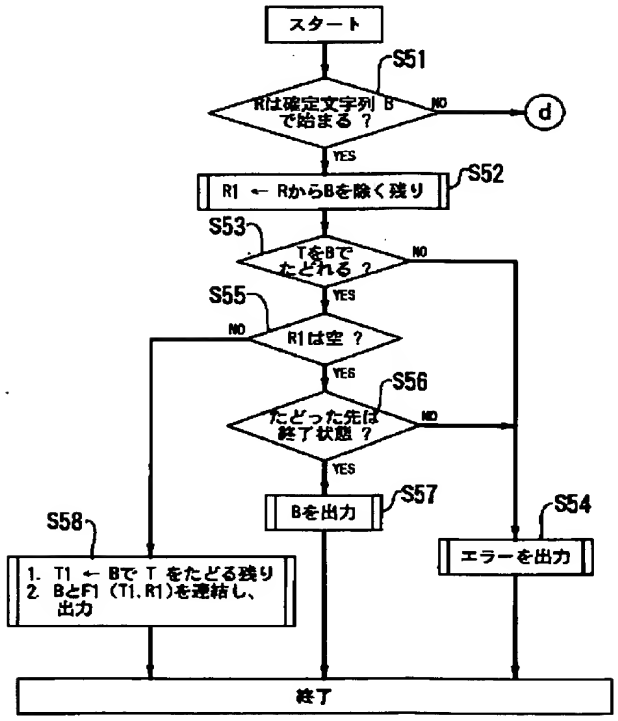
【図１３】



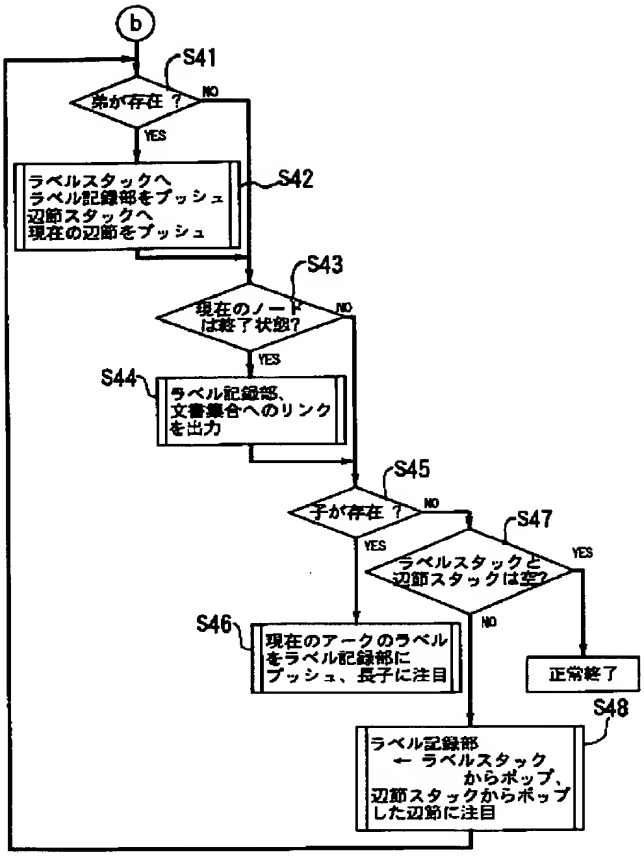
【図１５】



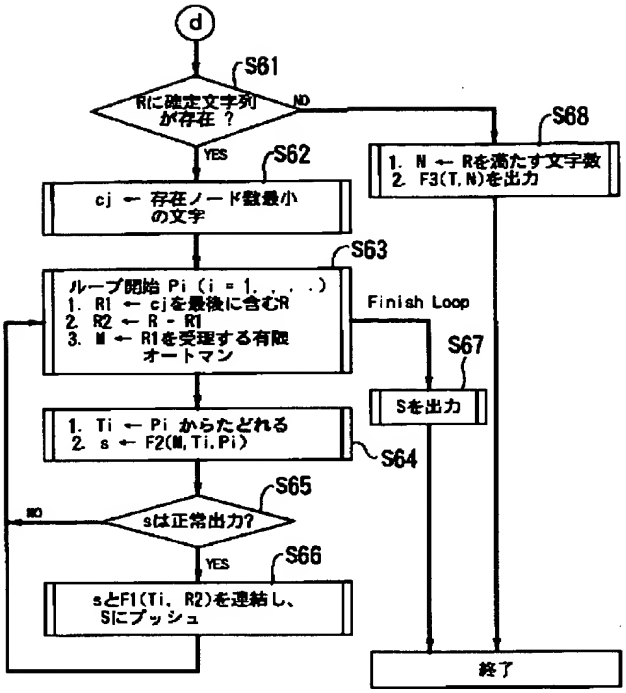
【図１６】



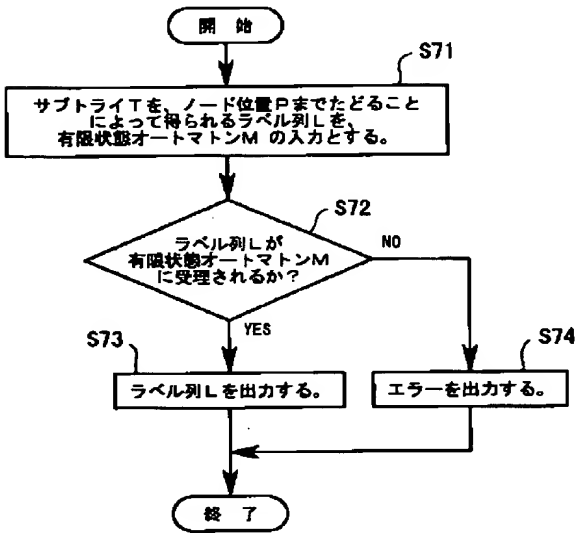
【図１４】



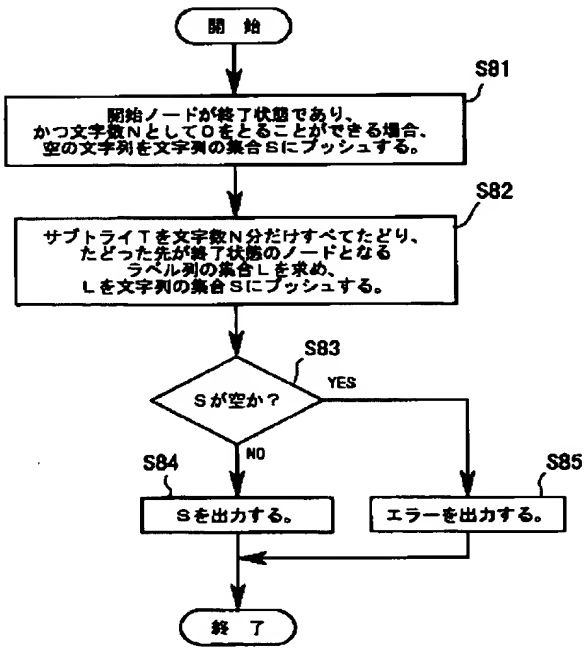
【図１７】



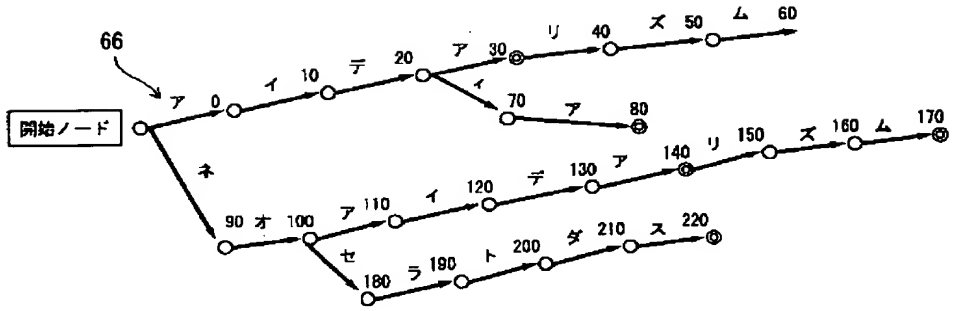
【図１８】



【図１９】



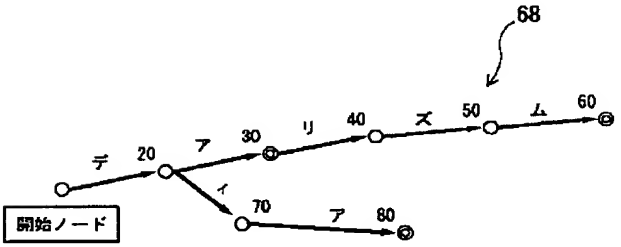
【図２０】



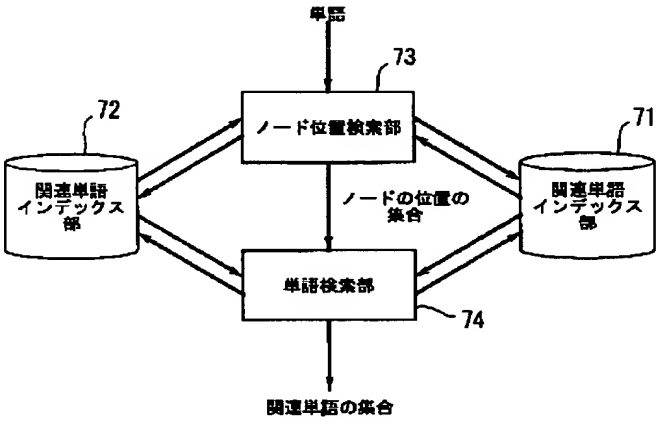
【図２１】

文字	出現数	対応するノード位置
ア	5	0, 30, 80, 110, 140
ィ	1	70
イ	2	10, 120
オ	1	100
ス	1	220
ズ	2	50, 160
セ	1	180
ダ	1	210
デ	2	20, 130
ト	1	200
ネ	1	90
ム	2	60, 170
ラ	1	190
リ	2	40, 150

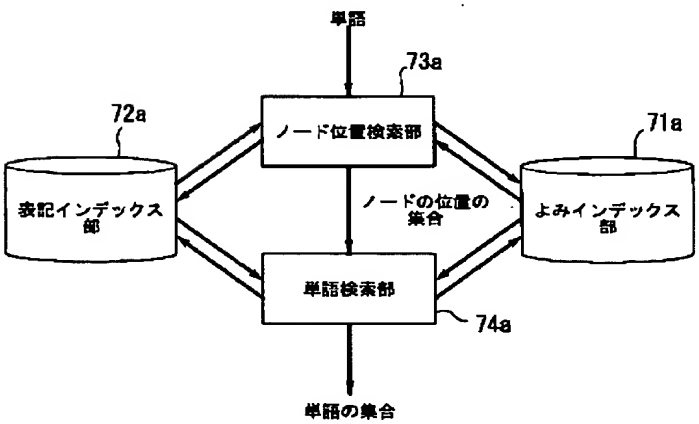
【図２３】



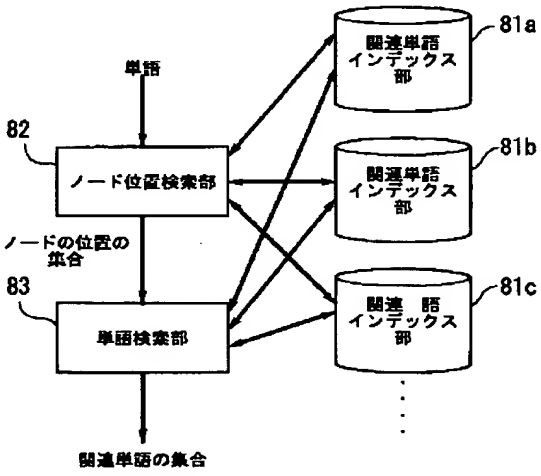
【図２４】



【図２５】



【図２６】



【図２７】

表記で表される単語	対応するよみ
A	あるふぁ、えー
ABC	えーびーしー
ABC兵器	えーびーしーへいき
AB型	えーびーがた
AM	えーえむ
AM放送	えーえむほうそう
AP	えーぴー
ASA感光度	あさかんこうど、えーえすえーかんこうど
・・・	・・・

【図２８】

よみで表される単語	対応する表記
あ	あ、ア、亜、阿、在、有
あーく	アーク
あーくとう	アーク灯、アーク燈
あーくらいと	アークライト
あーけーど	アーケード
あーけいっく	アーケイック
あーす	アース
あーち	アーチ
・・・	・・・

【図２９】

	表記単語	よみ単語
単語数	93,452	68,819
トライのサイズ	688,312.0バイト	569,267.0バイト
トライサイズの合計	1,257,579.0バイト	
ポインタのエントリ数	98817	98817
ポインタの幅	2.5バイト	2.5バイト
ポインタ・テーブルサイズ	247042.5バイト	247042.5バイト
ポインタ・テーブルサイズの合計	494,085.0バイト	
インデックスの総容量	1,751,664.0バイト	

【図３０】

	表記単語	よみ単語
単語数	93,452	68,819
トライのサイズ	688,312.0バイト	569,267.0バイト
トライサイズの合計	1,257,579.0バイト	
テキストのサイズ	978,775.0バイト	658,118.0バイト
テキスト参照位置テーブルの幅	2.5バイト	2.5バイト
テキスト参照位置テーブルのサイズ	233,630.0バイト	172,047.5バイト
テキストと参照テーブルの合計サイズ	2,042,570.5バイト	
ポインタの幅	2.5バイト	2.5バイト
ポインタのエントリ数	98817	98817
ポインタ・テーブルサイズ	247042.5バイト	247042.5バイト
ポインタ・テーブルサイズの合計	494,085.0バイト	
トライを除くインデックスの合計サイズ	2,536,655.5バイト	
インデックスの総容量	3,794,234.5バイト	

【図３１】

	表記単語	よみ単語
単語数	93,452	68,819
トライのノード数	254,184	230,783
予想される親ノードへのリンクサイズ	635460.0バイト	576,957.5バイト
予想されるトライサイズ	1,323,772.0バイト	1,146,224.5バイト
予想されるトライサイズの合計		2,469,996.5バイト
ポインタの幅	2.5バイト	2.5バイト
ポインタのエントリ数	98817	98817
ポインタ・テーブルサイズ	247042.5バイト	247042.5バイト
ポインタ・テーブルサイズの合計		494,085.0バイト
予想されるインデックスの総容量		2,964,081.5バイト

【図３２】

	従来技術 A を100 とした場合の容量比	従来技術 B を100 とした場合の予想容量比
トライのサイズ	100	50.9
トライを除くインデックスの合計サイズ	19.5	100
インデックスの総容量	46.2	59.1